

# Session Report



## Test & Feedback

Started On: 1/1/2023 07:24 PM  
Exported  
On: 1/1/2023 08:22 PM  
Bugs Filed: 3  
Time Zone: Eastern European Standard Time

This report was generated using [Test & Feedback extension](#) for [Azure DevOps Services](#) / Team Foundation Server in standalone mode (with limited functionality). Connect to Azure DevOps for the full featured version, learn more - [Test & Feedback Marketplace page](#).

## Summary of bugs filed

- 
- Bug 1** 1/1/2023 07:53 PM Dayane\_Lima\_DE\_Env: Paramenter "uptime" Null
- 
- Bug 2** 1/1/2023 08:06 PM Dayane\_Lima\_UK: Misspelling in the "status" parameter
- 
- Bug 3** 1/1/2023 08:14 PM Dayane\_Lima\_UK: The health" is "UNSTABLE"
- 

**Bug 1** 1/1/2023 07:53 PM

Dayane\_Lima\_DE\_Env: Paramenter "uptime" Null

**Test Scenario:** Validate Json data Type

**Environment:** Dayane\_Lima\_DE

**HTTPS Request:** GET <https://4s9rh46bpe.execute-api.eu-central-1.amazonaws.com/test/healthcheck>

```
pm.expect(jsonData.status.uptime).to.be.a("number");
```

**Actual Result:** The response payload returned "null" the parameter "uptime"

**Expected Result:** On the Dayane\_Lima\_UK\_Env the jason schema returns "integer"

The screenshot displays the Postman interface with the 'Dayane\_Lima' collection selected. The 'Run results' section shows a summary of the test run on 1/3/23 at 19:27:39. The summary table indicates 40 tests passed, 1 failed, and 0 skipped. The failed test is 'Data Type Validations | AssertionError: expected null to be a number'. A red arrow points to this failed test entry in the list.

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	Dayane_Lima_DE	1	1s 322ms	40	203 ms

**All Tests** Passed (39) Failed (1) Skipped (0) [View Summary](#)

- Pass Headers: x-amzn-Requestid is Present
- Pass Headers: Date is Present
- Pass Headers: Content-Length is Present
- GET Healthcheck** `{{baseUrl}}/healthcheck` / Healthcheck / Healthcheck 200 OK 201 ms 369 B
- Pass Headers: Content-Type is application/json
- Pass Headers: Connection is keep-alive
- Pass Headers: Content-Type is Present
- Pass Headers: x-amz-apigw-id is Present
- Pass Headers: x-amzn-Requestid is Present
- Pass Headers: Date is Present
- Pass Headers: Content-Length is Present
- Fail** Data Type Validations | AssertionError: expected null to be a number
- Pass Validate Response Data

Dayane\_Lima - Run results

Run on Today, 19:27:39 · View all runs

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	Dayane_Lima_DE	1	1s 322ms	40	203 ms

All Tests: Passed (39) Failed (1) Skipped (0)

View Summary

- Pass Headers: x-amzn-Requestid is Present
- Pass Headers: Date is Present
- Pass Headers: Content-Length is Present
- GET Healthcheck ((baseUri))/healthcheck / Healthcheck / Healthcheck 200 OK 201 ms 369 B
- Pass Headers: Content-Type is application/json
- Pass Headers: Connection is keep-alive
- Pass Headers: Content-Type is Present
- Pass Headers: x-amz-apigw-id is Present
- Pass Headers: x-amzn-Requestid is Present
- Pass Headers: Date is Present
- Pass Headers: Content-Length is Present
- Fail Data Type Validations | AssertionError: expected null to be a number
- Pass Validate Response Data

Dayane\_Lima / Healthcheck / Healthcheck

GET ((baseUri))/healthcheck

Params Authorization Headers (10) Body Pre-request Script Tests Settings

```
1 // Validate json data Type
2 const jsonData = pm.response.json();
3 pm.test("Data Type Validations", () => {
4   pm.expect(jsonData).to.be.a("object");
5   pm.expect(jsonData.market).to.be.a("string");
6   pm.expect(jsonData.status).to.be.a("object");
7   pm.expect(jsonData.status.health).to.be.a("string");
8   pm.expect(jsonData.status.uptime).to.be.a("number");
9 });
10 //Validate the data
11 pm.test("Validate Response Data", () => {
12   pm.expect(jsonData.status.health).to.eql("HEALTHY");
13 });
```

Body Cookies Headers (7) Test Results (8/9)

Status: 200 OK Time: 206 ms Size: 369 B Save Response

```
1 {
2   "apiVersion": "1.0.0",
3   "market": "DE",
4   "status": {
5     "health": "HEALTHY",
6     "uptime": null
7   }
8 }
```

## Json Schema

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "apiVersion": {
      "type": "string"
    },
    "market": {
      "type": "string"
    },
    "status": {
      "type": "object",
      "properties": {
        "health": {
          "type": "string"
        },
        "uptime": {
          "type": "null"
        }
      },
      "required": [
        "health",
        "uptime"
      ]
    },
    "required": [
      "apiVersion",
      "market",
      "status"
    ]
  }
}
```

**Bug 2** 1/1/2023 08:06 PM

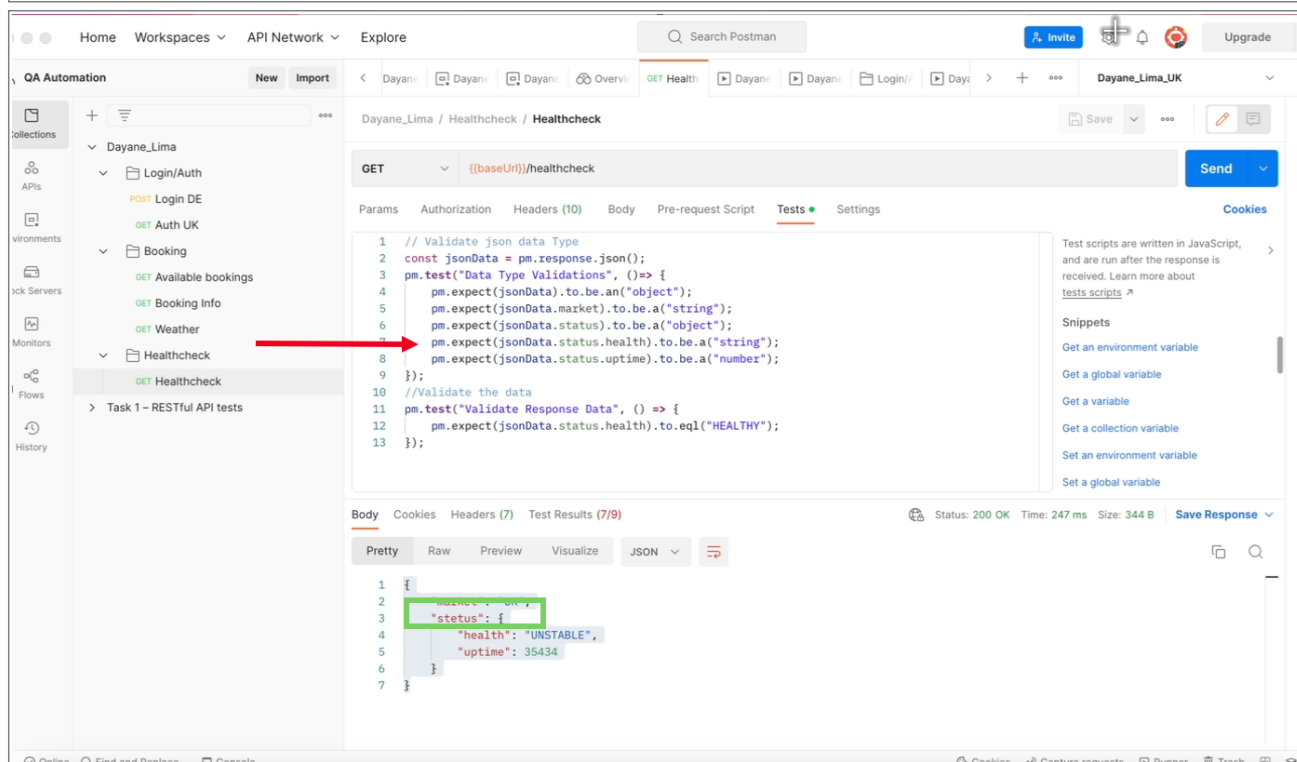
Dayane\_Lima\_UK: Misspelling in the "status" parameter

---

**Test Scenario: Validate Json Response Body****Environment:** Dayane\_Lima\_UK**HTTPS Request:** GET <https://4s9rh46bpe.execute-api.eu-central-1.amazonaws.com/test/healthcheck>

Failed test:

```
pm.expect(jsonData.status).to.be.a("object");  
pm.expect(jsonData.status.health).to.be.a("string");  
pm.expect(jsonData.status.uptime).to.be.a("integer");
```

**Actual Result:** On the response payload - Misspelling in the "stetus" parameter**Expected Result:** Response payload must display "Status"**Bug 3** 1/1/2023 08:14 PM

Dayane\_Lima\_UK: The health" is "UNSTABLE"

**Test Scenario: Validate Json Response Body****Environment:** Dayane\_Lima\_UK**HTTPS Request:** GET <https://4s9rh46bpe.execute-api.eu-central-1.amazonaws.com/test/healthcheck>

Failed test:

```
pm.test("Validate Response Data", () => {  
    pm.expect(jsonData.status.health).to.eql("HEALTHY");  
});
```

**Actual Result:** The The health" is "UNSTABLE"**Expected Result:** The Dayane\_Lima\_UK must be "health"

The screenshot displays the Postman interface for a REST client. The left sidebar shows a collection named 'Dayane\_Lima' with a sub-collection 'Healthcheck'. The main panel shows a GET request to '({baseUrl})/healthcheck'. The 'Tests' tab is active, showing a JavaScript test script. A green box highlights the test script, and a red arrow points to the 'health' property in the JSON response body, which is 'UNSTABLE'.

**Test Script (JavaScript):**

```
1 // Validate json data Type  
2 const jsonData = pm.response.json();  
3 pm.test("Data Type Validations", ()=> {  
4     pm.expect(jsonData).to.be.an("object");  
5     pm.expect(jsonData.market).to.be.a("string");  
6     pm.expect(jsonData.status).to.be.a("object");  
7     pm.expect(jsonData.status.health).to.be.a("string");  
8     pm.expect(jsonData.status.uptime).to.be.a("number");  
9 }  
10  
11 //Validate the data  
12 pm.test("Validate Response Data", () => {  
13     pm.expect(jsonData.status.health).to.eql("HEALTHY");  
14 });
```

**JSON Response Body:**

```
{  
  "market": "UK",  
  "status": {  
    "health": "UNSTABLE",  
    "uptime": 35434  
  }  
}
```

## Session attachments

Screenshot-1.png 1/1/2023 07:28 PM

The screenshot displays the Postman QA Automation interface. The left sidebar shows a tree view of collections under 'QA Automation', with 'Dayane\_Lima' expanded. The main panel shows the 'Run results' for 'Dayane\_Lima', including a summary table and a detailed list of test results.

**Dayane\_Lima - Run results**

Run on Today, 19:27:39 · [View all runs](#)

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	Dayane_Lima_DE	1	1s 322ms	40	203 ms

**All Tests**   Passed (39)   Failed (1)   Skipped (0)

- Pass Headers: x-amzn-Requestid is Present
- Pass Headers: Date is Present
- Pass Headers: Content-Length is Present
- GET Healthcheck** {{baseUrl}}/healthcheck / Healthcheck / Healthcheck 200
  - Pass Headers: Content-Type is application/json
  - Pass Headers: Connection is keep-alive
  - Pass Headers: Content-Type is Present
  - Pass Headers: x-amz-apigw-id is Present
  - Pass Headers: x-amzn-Requestid is Present
  - Pass Headers: Date is Present
  - Pass Headers: Content-Length is Present
  - Fail Data Type Validations | AssertionError: expected null to be a number**
  - Pass Validate Response Data

A red arrow points to the failed test result: "Data Type Validations | AssertionError: expected null to be a number".

Screenshot-2.png 1/1/2023 07:29 PM

QA Automation

Home Workspaces API Network Explore

Search Postman

invite

Dayane\_Lima

Run Again Automate Run

Dayane\_Lima - Run results

Run on Today, 19:27:39 · View all runs

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	Dayane_Lima_DE	1	1s 322ms	40	203 ms

All Tests Passed (39) Failed (1) Skipped (0)

Pass Headers: x-amzn-RequestId is Present

Pass Headers: Date is Present

Pass Headers: Content-Length is Present

GET Healthcheck {{baseUrl}}/healthcheck / Healthcheck / Healthcheck 200 OK

Pass Headers: Content-Type is application/json

Pass Headers: Connection is keep-alive

Pass Headers: Content-Type is Present

Pass Headers: x-amz-apigw-id is Present

Pass Headers: x-amzn-RequestId is Present

Pass Headers: Date is Present

Pass Headers: Content-Length is Present

Fail Data Type Validations | AssertionError: expected null to be a number

Pass Validate Response Data



Screenshot-3.png 1/1/2023 07:31 PM

The screenshot displays the Postman application interface. The left sidebar shows the 'QA Automation' workspace with a collection named 'Dayane\_Lima' containing several API endpoints. The main panel shows a REST client request for the 'Healthcheck' endpoint. The 'Tests' tab is active, showing the following JavaScript code:

```
1 // Validate json data Type
2 const jsonData = pm.response.json();
3 pm.test("Data Type Validations", ()=> {
4     pm.expect(jsonData).to.be.an("object");
5     pm.expect(jsonData.market).to.be.a("string");
6     pm.expect(jsonData.status).to.be.a("object");
7     pm.expect(jsonData.status.health).to.be.a("string");
8     pm.expect(jsonData.status.uptime).to.be.a("number");
9 });
10 //Validate the data
11 pm.test("Validate Response Data", () => {
12     pm.expect(jsonData.status.health).to.eql("HEALTHY");
13 });
```

The 'Body' tab shows the JSON response:

```
1 {
2   "apiVersion": "1.0.0",
3   "market": "DE",
4   "status": {
5     "health": "HEALTHY",
6     "uptime": null
7   }
8 }
```

The status bar at the bottom indicates a successful response with status 200 OK, time 206 ms, and size 36.

**Note-1** 1/1/2023 07:34 PM

Json Schema { "\$schema": "http://json-schema.org/draft-04/schema#", "type": "object", "properties": { "apiVersion": { "type": "string" }, "market": { "type": "string" }, "status": { "type": "object", "properties": { "health": { "type": "string" }, "uptime": { "type": "null" } }, "required": [ "health", "uptime" ] } }, "required": [ "apiVersion", "market", "status" ] }

**Screenshot-7.png** 1/1/2023 07:54 PM

Home Workspaces API Network Explore

Search Postman

QA Automation

New Import

Dayane Dayane Dayane Overview GET Health Dayane Dayane Login// Dayane

Dayane\_Lima

Save

Dayane\_Lima / Healthcheck / Healthcheck

GET {{baseUrl}}/healthcheck

Params Authorization Headers (10) Body Pre-request Script Tests Settings

```
1 // Validate json data Type
2 const jsonData = pm.response.json();
3 pm.test("Data Type Validations", () => {
4   pm.expect(jsonData).to.be.an("object");
5   pm.expect(jsonData.market).to.be.a("string");
6   pm.expect(jsonData.status).to.be.a("object");
7   pm.expect(jsonData.status.health).to.be.a("string");
8   pm.expect(jsonData.status.uptime).to.be.a("number");
9 });
10 //Validate the data
11 pm.test("Validate Response Data", () => {
12   pm.expect(jsonData.status.health).to.eql("HEALTHY");
13 });
```

Test scripts are written and are run after the response is received. Learn more [tests scripts](#)

Snippets

- Get an environment variable
- Get a global variable
- Get a variable
- Get a collection variable
- Set an environment variable
- Set a global variable

Body Cookies Headers (7) Test Results (7/9) Status: 200 OK Time: 247 ms Size: 344 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "market": "UK",
3   "stetus": {
4     "health": "UNSTABLE",
5     "uptime": 35434
6   }
7 }
```

Screenshot-8.png 1/1/2023 07:57 PM

The screenshot displays the Postman interface for a REST client. The left sidebar shows a collection named 'Dayane\_Lima' with a folder 'Healthcheck' containing a 'GET Healthcheck' request. A red arrow points to this request. The main panel shows the details of the 'GET Healthcheck' request, including the URL 'GET {{baseUrl}}/healthcheck' and the 'Tests' tab. The test scripts are as follows:

```
1 // Validate json data Type
2 const jsonData = pm.response.json();
3 pm.test("Data Type Validations", () => {
4   pm.expect(jsonData).to.be.an("object");
5   pm.expect(jsonData.market).to.be.a("string");
6   pm.expect(jsonData.status).to.be.a("object");
7   pm.expect(jsonData.status.health).to.be.a("string");
8   pm.expect(jsonData.status.uptime).to.be.a("number");
9 });
10 //Validate the data
11 pm.test("Validate Response Data", () => {
12   pm.expect(jsonData.status.health).to.eql("HEALTHY");
13 });
```

The 'Body' tab shows the response in JSON format:

```
1 {
2   "market": "OK",
3   "status": {
4     "health": "UNSTABLE",
5     "uptime": 35434
6   }
7 }
```

The status bar at the bottom indicates 'Status: 200 OK', 'Time: 247 ms', and 'Size: 344 B'.

Screenshot-9.png 1/1/2023 08:08 PM

The screenshot displays the Postman interface for a GET request to `{{baseUrl}}/healthcheck`. The left sidebar shows the 'Collections' panel with a tree structure: Dayane\_Lima > Login/Auth > POST Login DE, GET Auth UK, Booking > GET Available bookings, GET Booking Info, GET Weather, Healthcheck > GET Healthcheck. The main panel shows the 'Tests' tab with the following JavaScript code:

```
1 // Validate json data Type
2 const jsonData = pm.response.json();
3 pm.test("Data Type Validations", ()=> {
4   pm.expect(jsonData).to.be.an("object");
5   pm.expect(jsonData.market).to.be.a("string");
6   pm.expect(jsonData.status).to.be.a("object");
7   pm.expect(jsonData.status.health).to.be.a("string");
8   pm.expect(jsonData.status.uptime).to.be.a("number");
9 });
10 //Validate the data
11 pm.test("Validate Response Data", () => {
12   pm.expect(jsonData.status.health).to.eql("HEALTHY");
13 });
```

The 'Body' tab shows the JSON response:

```
1 {
2   "market": "UK",
3   "stetus": {
4     "health": "UNSTABLE",
5     "uptime": 35434
6   }
7 }
```

A red arrow points to the `"health": "UNSTABLE"` value in the response. The status bar at the bottom indicates 'Status: 200 OK', 'Time: 247 ms', and 'Size: 344 B'.

Screenshot-10.png 1/1/2023 08:18 PM

The screenshot shows the Postman interface with a GET request selected. The URL is `{{baseUrl}}/auth/login?bookingRef={{bookingRef}}&surname={{surname}}`. The parameters section shows `bookingRef` and `surname` with values `{{bookingRef}}` and `{{surname}}` respectively. The response body is displayed in JSON format, showing `"bookingRef": "AR58930"` and `"surname": "Smith"`. A red arrow points to the `Pre-request Script` tab.

QA Automation

New Import

Dayane\_Lima / Login/Auth / Auth UK

GET `{{baseUrl}}/auth/login?bookingRef={{bookingRef}}&surname={{surname}}`

Params Authorization Headers (11) Body Pre-request Script Tests Settings

☒ x-api-key `0qNSzqieHI5M6LLgY7Y85Zco0SMXKyr37MIOSAG`

☒ bookingRef `{{bookingRef}}`

☒ surname `{{surname}}`

Key Value Description

Body Cookies Headers (8) Test Results (10/11) Status: 200 OK Time: 215 ms Size: 337 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "bookingRef": "AR58930",
3   "surname": "Smith"
4 }
```

**Note-2** 1/1/2023 08:22 PM

Environment: Dayane\_Lima\_UK HTTPS Request: GET <https://4s9rh46bpe.execute-api.eu-central-1.amazonaws.com/auth/login?bookingRef=AR58930&surname=Smith> The User Authentication information is sent in the URL

## End of report.

---

Want to track bugs in a more seamless manner? Connect to [Azure DevOps](#) for free.